

# How to get Cross Sections from MIPP data

Holger Meyer

March 8, 2009

## Abstract

This document describes the algorithms for extracting cross sections from reconstructed MIPP data. We discuss the trigger and the scalars as they relate to cross sections.

## 1 Introduction

The probability of an interaction between a beam particle and a target particle is given by the *cross section* for the process in question. The probability is the cross section area of the target normalized by the area the beam is spread over. Of course the cross sections in particle physics are determined by the forces between the interacting particles rather than geometric cross section areas of the particles.

### 1.1 Cross section terminology

The *exclusive* cross section gives the probability of an exclusive final state, e.g.  $pp \rightarrow p\pi^+n$ . *Inclusive* cross sections give the probability of observing some final state particles together with anything else, e.g.  $pp \rightarrow \pi^+X$  where the  $X$  represents anything else. The ratio of two single particle inclusive cross sections is production ratio for these particles and is simpler to determine experimentally. We usually measure differential cross sections  $\frac{d\sigma}{d\Omega}$  where  $d\Omega$  is the solid angle that a final state particle moves to. The differential cross section is a function of the polar angle  $\theta$ , but (due to rotational symmetry around the beam axis) it does not depend on the azimuthal angle  $\phi$  (unless beam or target are polarized).

The total or integral cross section is the integral of the differential cross section over the full solid angle,  $\sigma = \oint_{4\pi} \frac{d\sigma}{d\Omega} d\Omega$ .

Cross sections can be expressed differential with respect to other quantities relating to the final state than the solid angle. The differential cross section  $\frac{d\sigma}{dp}$  versus momentum of a final state particle is an example. Double differential cross sections express the cross section as a function of two variables.

Cross sections are measured in units of *barn* ( $1b = 10^{-28}m^2 = 10^{-24}cm^2$ ).

## 2 Measurement of cross sections

To measure cross sections we have to determine the number of interactions with the desired final state and the number of incident beam particles. The cross section is determined from experimental data as follows:

$$\frac{d\sigma}{d\Omega} = \frac{dN_{int}}{N_{beam} \cdot n_{tgt} \cdot d\Omega} \quad (1)$$

In this equation

- $dN_{int}$  is the number of interactions with a final state particle moving into a solid angle  $d\Omega$  (or having momentum  $dp$  or transverse momentum  $dp_t$  or whichever other differential variable characterizing the final state)
- $N_{beam}$  is the number of incident beam particles
- $n_{tgt} = \frac{N_A}{V_{mol}} \cdot L_{tgt}$  is the areal target density, the number of target particles per area ( $N_A$  is Avogadro's number,  $V_{mol}$  is the molar volume, molar mass divided by density), and  $L_{tgt}$  is the target length along the beam direction

Ideally the experiment would send beam at the target one particle at a time when the detector is ready to read out the event. The number of beam particles would be counted directly and the number of interactions would be determined from the recorded events by applying corrections for the detection efficiency and geometric acceptance of the final state under consideration. However, for small cross sections it is not practicable to perform the experiment in this way because it would take too long to acquire sufficient statistics.

Instead a steady beam of particles is send onto the target and event readout is triggered. Scalers count the number of incident beam particles. The total beam flux is larger than the number determined by the scaler because two particles may pass through the beam detection device (the T0 counters in MIPP) simultaneously, leaving only one recordable signal. This can be corrected for in the data analysis. To keep the correction small the beam intensity should be small compared to the minimum separation between two hits required by the scaler. In MIPP the TBD and T01 counters and their coincidence (the *beam* signal) are all scaled and also latched for each event.

### 2.1 Dead-time

The readout of an interaction event in the detector takes time. During this time the trigger is self-inhibited because the ADCs and TDCs are not capable of recording new data until the previous event has been read out and the electronics has been cleared. During this time the detector is blind to new interactions. This time is called dead-time. The remaining time (when the detector is waiting for the next trigger) is the live-time.

Dead-time can be minimized or practically eliminated using *dead-time less* detector readout schemes. The MIPP upgrade will take steps in this direction. However, for a true 100% live-time fraction the TPC would need to be sensitive all the time. This is not possible. The TPC requires

the gating grid to prevent space charge build-up and breakdown in the amplification drift region near the anode wires. Thus the MIPP detector will always have some dead-time.

The dead-time is measured by scaling an rf for the duration of interest (a beam slow-spill in MIPP) and scaling the same rf during those times when the trigger is inhibited, i.e. scaling the rf gated by the trigger inhibit signal.

To determine cross sections either the number of recorded events has to be scaled up from the live-time fraction to 1 or the entire beam flux has to be scaled down by the live-time fraction. The second approach is preferable.

### 3 Scaler counts and trigger information in MIPP DSTs

The MIPP DSTs contain three separate TTrees in a single ROOT file. These trees contain information on the run, the spills, and the events. The corresponding classes are MIPPRunSummary, MIPPSpillSummary, and MIPPEventSummary, respectively.

#### 3.1 Run Summary

A DST usually contains a single MIPPRunSummary with global run information. The data member of interest here is

```
int trigps[32];          ///< Trigger prescales
```

This array contains the trigger prescale values for each of the 20 trigger bits.

#### 3.2 Spill Summary

The spill-tree contains a MIPPSpillSummary object for each spill in the event. The class contains the following public data members of interest here:

```
int run;                  ///< Run number
int spill;                ///< spill number
int nevt;                 ///< number of events

int rawTrigInput[40];     ///< raw trigger scaler count
int gatedTrigBit[20];     ///< gated trigger bit scaler count
int rawTrigBit[20];       ///< raw trigger bit scaler count
int psTrigBit[20];        ///< prescaled trigger bit scaler count
int psAndGatedTrigBit[20]; ///< prescaled and gated trigger bit scaler count
```

The spill number *spill* is sequential and is also contained in each MIPPEventSummary to link an event with the spill information. Thus each spill can be considered a mini-subrun. When processing DSTs cuts can be placed on the slow monitoring alarm information provided in the

spill summaries to cut bad spills. Then all events in these spills also have to be cut based on their spill number.

The number of events *nevt* is the number of events in this beam spill. The sum of *nevt* over all spill summaries in a run is less than the number of events given in the run summary because out-of-spill calibration events are not counted in the spill summaries.

The four arrays of 20 integers contain the scaler counts for each trigger bit with and without the live-gate applied and before and after applying the prescale set in the run. Trigger bits that have been disabled will have prescaled scaler counts of 0. The gated counts will be lower than the corresponding raw counts. Dead-time can be determined from any of the 40 ratios of gated trigger bits to their corresponding ungated trigger bits. The 20 ratios of prescaled scalers have low to no statistics. The other twenty ratios will each give slightly different results. Consider for example the time at the beginning of a spill when the detector is live until the first event in the spill is triggered. Both raw and gated rf scalers will increase their counts, but none of the interaction trigger scalers will increment until the first interaction in the spill is triggered. Taking these variations into account, it is useful to look at the '*interaction trigger life-time*' for diagnostic purposes. The life-time relevant for cross section determinations is the ratio of the un-prescaled raw (ungated) and gated rf scalers. The rf (a scaled down copy of the Main Injector rf) was called *Raw Pulser* and was connected to bit 12 in the trigger throughout the run, i.e. the 13<sup>th</sup> bit counting from 0, `rawTrigBit[13]` and `gatedTrigBit[13]`. The prescale for this bit was always set to -1 (disabled).

The array *rawTrigInput* contains numbers scaled at the input to the trigger. It is for diagnostic use and not needed for cross section determinations.

### 3.3 Event Summary

The `MIPPEventSummary` class contains the following data members for each event:

```
int spill;                ///< spill number

Short_t rawtrig;          ///< raw trigger word
Short_t pstrig;           ///< prescale trigger word
Short_t rf_tdc[16];
```

The variable *spill* links the event to the corresponding spill summary, allowing for cuts to be placed on a spill-by-spill basis on e.g. the slow-monitoring alarm information (TPC HV trips, etc.) in the spill summaries.

The other data members of interest here are the raw and prescaled trigger words. The term 'word' here is used in the computer science meaning of 'data word', i.e. an integer encoded in a certain number of bits; e.g. a byte is a 8-bit data word. To test if a bit *i* is set in a given event, mask the trigger word with a word that contains all zeros with a 1 at the *i*<sup>th</sup> position. In C++ the code fragment is

```
if(evt->rawtrig && (1<i)) {bit_i = true;}  
else {bit_i = false;}
```

The `rf_tdc` contains the rf hits during this event. It allows to determine the phase of the event relative to the Main Injector rf. It is not needed for cross section determinations.

## 4 Putting it all together

—To be done— (See macros in `DSTAnalysis...`)

### 4.1 Empty target subtraction

### 4.2 Finite bin size correction

While the cross section formula contains differential quantities, we actually measure over a small extended range. A correction may need to be applied if the yield is non-uniform over the range  $\Delta\Omega$  (or  $\Delta p$ ,  $\Delta p_T, \dots$ ).